H8 Simple Simulator

H8SS

H8/3664F 簡易シミュレータ

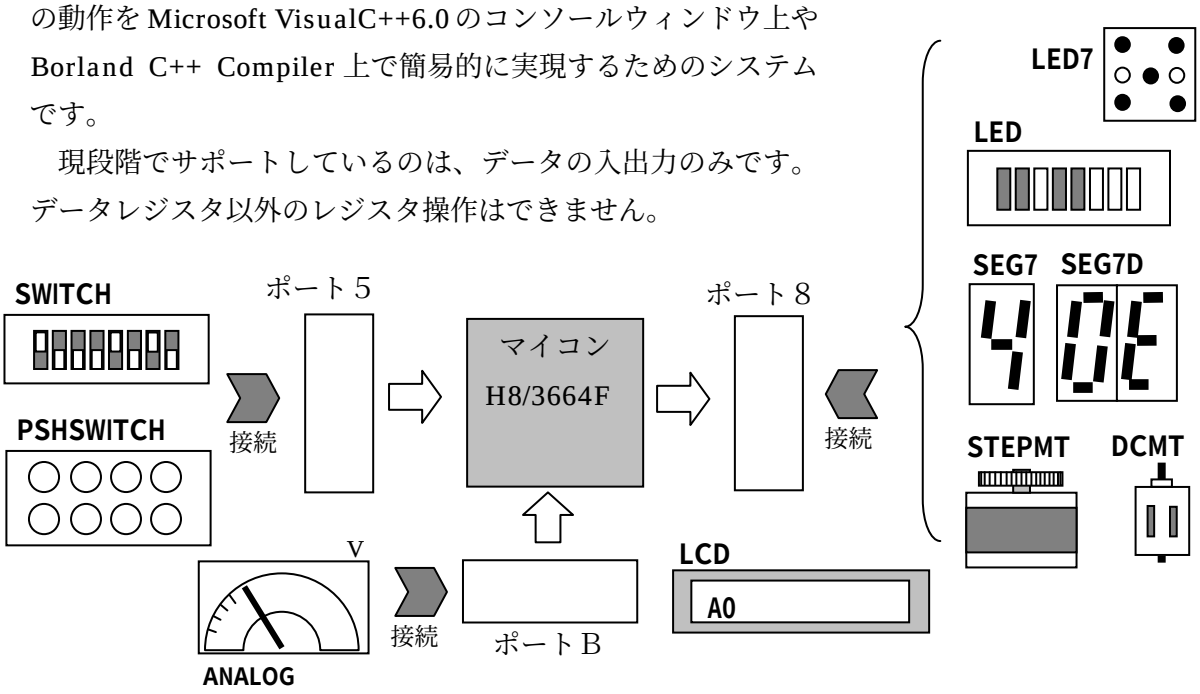
取扱説明書

Ver1.35

１．概要

ＨＳＳは１６ビットマイコンの㈱ﾙﾈｻｽﾃｸﾉﾛｼﾞの H8/3664F の動作を Microsoft VisualC++6.0 のコンソールウィンドウ上や Borland C++ Compiler 上で簡易的に実現するためのシステムです。

現段階でサポートしているのは、データの入出力のみです。データレジスタ以外のレジスタ操作はできません。



２．使用可能ポートと利用可能ディバイス

ポート	用途	利用可能ディバイス
ポート５	８ビット入力専用ポート	**SWITCH,PSHSWITCH, VMIN**※
ポート８	８ビット出力専用ポート	**LED, LED7, SEG7, SEG7D, STEPMT, DCMT, VMOUT**※
ポートＢ	４チャンネル アナログ入力専用ポート	**ANALOG**
その他	ＬＣＤ表示	

※ＶＭＩＮ，ＶＭＯＵＴの詳細については、別冊の説明書を参照してください。

３．プログラムへの組み込みにおける注意（後で詳細に説明します）

（１） ＨＳＳへのデータの受け渡し処理が随所で必要です。
データレジスタとＨＳＳとのデータ受け渡しを、指定のタイミングで行う必要があります。

（２） 実行後に、各ポートの接続ディバイスを変更することができますが、
明示的にＨＳＳの初期化をすることもできます。

（３） 実際には行うべき細部の手続き（ポート設定など）は省略されています。

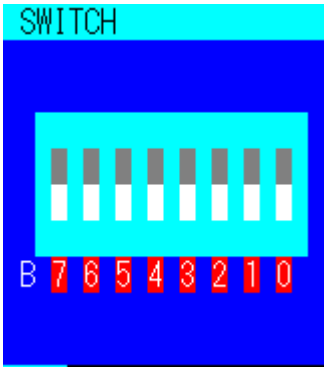
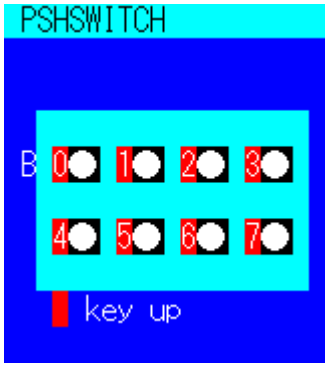
```
#include "3664.h"
void main(void)
{
   InitH8SS(NONE,LED,NONE);   （２）任意
    IO.PDR8.BYTE=0x55; io;    （１）必須
    while(1);
```


©2011 uaubn

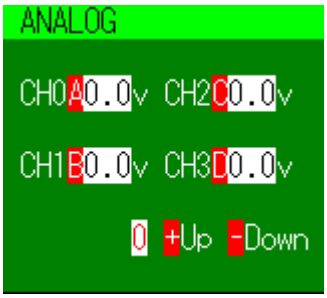
}

４．接続可能ディバイス一覧

Ｐ５接続

ＳＷＩＴＣＨ（８ビットスイッチ）	**ＰＳＨＳＷＩＴＣＨ（８ビットスイッチ）**
	

ＰＢ接続

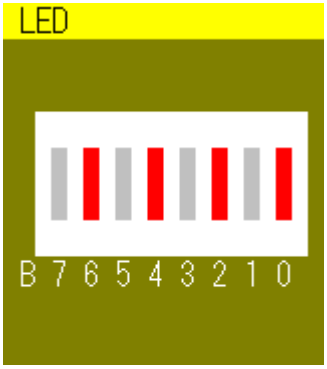
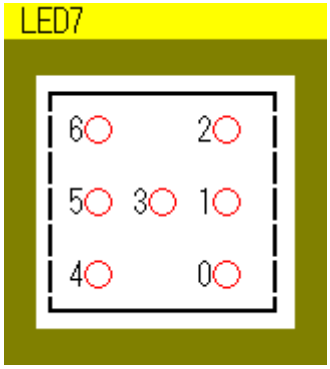
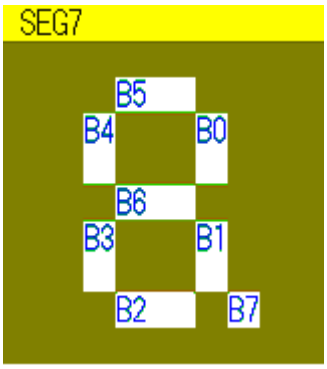
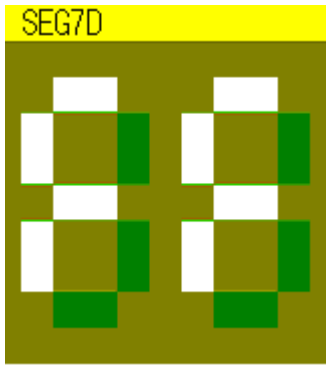
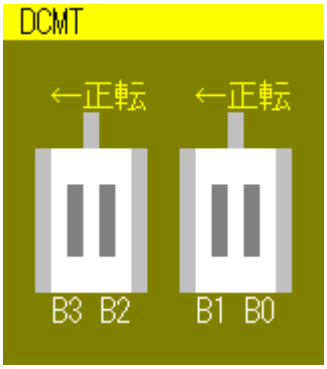
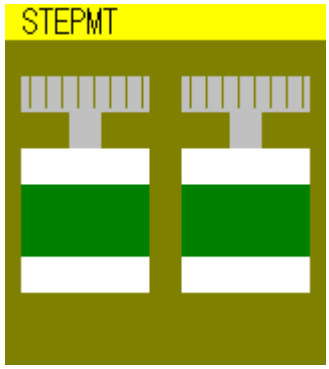
ＡＮＡＬＯＧ（４チャンネルアナログ入力）	
	

その他

ＬＣＤ（１６桁２行キャラクタディスプレイ）	

©2011 uaubn

Ｐ８接続

ＬＥＤ（８ビットＬＥＤ）	ＬＥＤ７（サイコロ状ＬＥＤ）
	
ＳＥＧ７（１桁７セグメント）	**ＳＥＧ７Ｄ（２桁７セグメント・デコーダ付）**
	
ＤＣＭＴ（ＤＣモータ２個）	**ＳＴＥＰＭＴ（ステッピングモータ２個）**
	

©2011 uaubn

５．Ｈ８ＳＳでのプログラム開発の方法

（１）データレジスタの内容とＨ８ＳＳとのデータの「受け渡し処理」を必要なタイミングで行ってください。Ｈ８ＳＳは、

①Ｈ８／３６６４のデータレジスタを画面に反映させる。

②画面での入力の状況をデータレジスタに反映させる。

ことを自動的に処理できないため、プログラム中でプログラマがデータの「受け渡し処理」を行うタイミングを指示する必要があります。

たとえば、スイッチの状況を取り出す場合、その値は、最後に「受け渡し処理」を実行した時の値となります。したがって、データレジスタの値を取り出す場合は、直前で「受け渡し処理」を行うことが理想的な状況となります。同様に、データレジスタにデータを出力する場合、その後に「受け渡し処理」が無い場合は、各ディバイスにその値は反映されません。

形式	**io;**
機能	Ｈ８ＳＳとデータレジスタの「受け渡し処理」を行います。 画面でのキー入力の取り込みと、データレジスタのディバイスへの出力処理を順に行います。
使用例	**IO.PDR8.BYTE=0x33; io; （出力値の反映）** **io; c=IO.PDR5.BIT.B0; （入力値の反映）** **io; IO.PDR8.BYTE=IO.PDR5.BYTE; io; （入力値と出力値の反映）**

なお、io は、#define io IOH8SS()として定義されています。

（２）Ｈ８ＳＳでの各ポートへ接続するディバイスを InitH8SS() 関数でも設定できます。

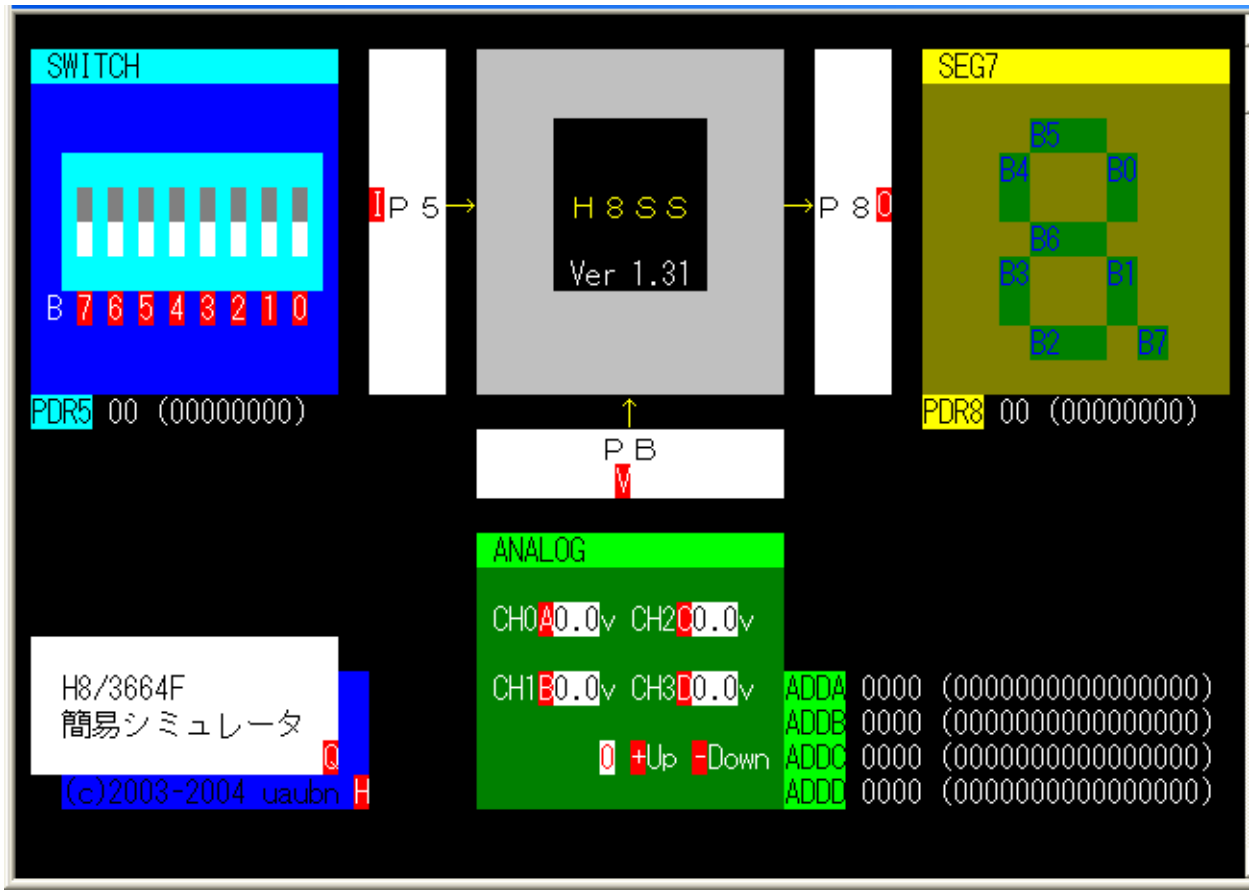
関数仕様

形式	**Void InitH8SS(int P5Device, int P8Device, int PBDevice)**
機能	Ｈ８ＳＳでの各ポートへ接続するディバイスを設定します。
引数	**P5Device:** ポート５に接続するディバイス指定 **P8Device:** ポート８に接続するディバイス指定 **PBDevice:** ポートＢに接続するディバイス指定
指定ディバイス	（ポート５用）**SWITCH** （ポート８用）**LED, LED7, SEG7, SEG7D, STEPMT, DCMT, LED7** （ポートＢ用）**ANALOG** （各ポート兼用）**NONE：**接続無し、
使用例	**InitH8SS(SWITCH,SEG7,NONE);** スイッチと７セグメントを接続 **InitH8SS(NONE,STEPMT,NONE);** ステッピングモータのみ接続 **InitH8SS(NONE,SEG7,ANALOG);** アナログ入力と７セグメントを接続

©2011 uaubn

６．Ｈ８ＳＳ操作方法

(1) 実行画面



(2) キー操作

操作キー	対象ディバイス	動作
I	**Ｐ５ディバイス**	Ｐ5に接続する入力ディバイスを変更します。
O	**Ｐ８ディバイス**	Ｐ8に接続する出力ディバイスを変更します。
V	**ＰＢディバイス**	ＰＢに接続するアナログ入力ディバイスを変更します。
スペース	**ＰＳＨＳＷＩＴＣＨ**	押したスイッチを戻します。
0 ～ 7	**ＳＷＩＴＣＨ** **ＰＳＨＳＷＩＴＣＨ**	スイッチのビットのＯＮ、ＯＦＦを行います。 **(0:B0, 1:B1, ・・・, 7:B7)**
A ～ D	**ＡＮＡＬＯＧ**	入力電圧を変更するチャンネルを変更します。 **(A:CH0, B:CH1, C:CH2, D:CH3)**
+, −		Ａ～Ｄ で選択したチャンネルの電圧(0～5Ｖ)の増減を行います。**(+: +0.5V, -: -0.5V)**
H	**Ｈ８ＳＳ**	サポート情報の表示
Q	**Ｈ８ＳＳ**	Ｈ8ＳＳの終了

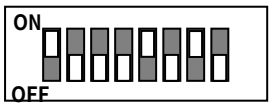
実行画面で、赤地に白字の文字が操作キーとなります。

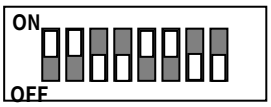
©2011 uaubn

７．各ディバイスの特徴

８桁スイッチ(SWITCH)

スイッチがＯＮの場合ポート５のビットが１に、ＯＦＦの場合ビットが０になります。





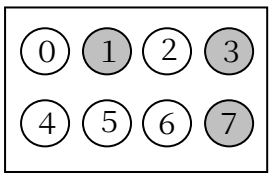
Port５

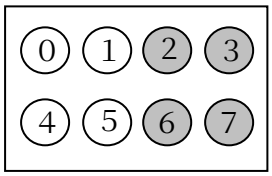
1	0	0	0	1	0	1	0
b7	b6	b5	b4	b3	b2	b1	b0

1	1	0	0	1	1	0	0
b7	b6	b5	b4	b3	b2	b1	b0

８桁スイッチ(PSHSWITCH)

スイッチがＯＮの場合ポート５のビットが０に、ＯＦＦの場合ビットが１になります。
スイッチの押している状態を解除するには、再度スイッチの番号を押してください。
（なお、スペースキーを押すことで、全部のキーをまとめて押していない状態にできます。）





Port５

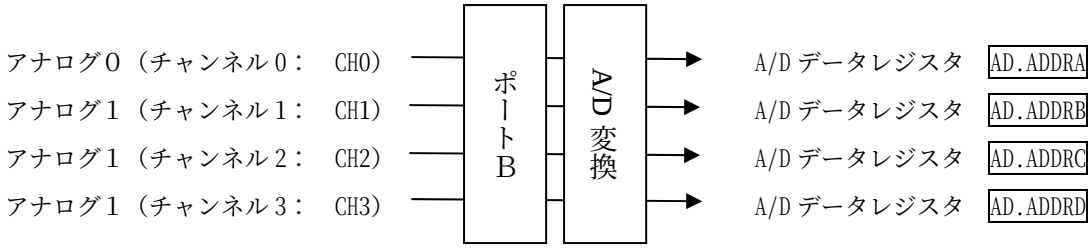
1	0	0	0	1	0	1	0
b7	b6	b5	b4	b3	b2	b1	b0

1	1	0	0	1	1	0	0
b7	b6	b5	b4	b3	b2	b1	b0

©2011 uaubn

アナログ値の入力(ANALOG)

アナログ値の入力

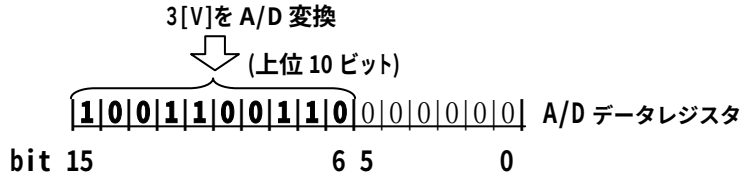


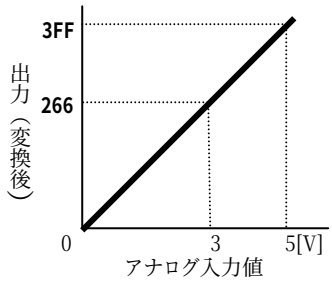
ポートBの各チャンネルの電圧をA/D（アナログ→デジタル）変換した10ビットの値が、A/D データレジスタに設定されます。

（例）アナログ入力値 → 変換値（10 ビット）
０Ｖ → 0
３Ｖ → 614(10)、1001100110(2)、266(16)
５Ｖ → 1023(10)、1111111111(2) 、3FF(16)

なお、A/D データレジスタは16ビットのレジスタで、変換後の10ビットは**上位10ビットに設定されます**（下位6ビットは0）。

今回は、4チャンネルのアナログ入力がポートBの下位4ビットにされているものとし、各変換後のデータは、自動的に、4つの A/D データレジスタ（AD.ADDRA, AD.ADDRB, AD.ADDRC, AD.ADDRD）に入力されているとします。



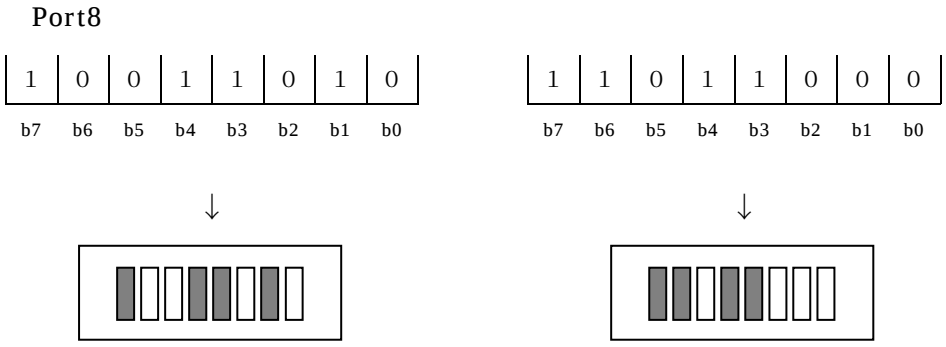


(注)H8SS では、A/D データレジスタ=(入力されたアナログ値/0.00488758553)<<6 で算出しています。

©2011 uaubn

8桁ＬＥＤ(LED)

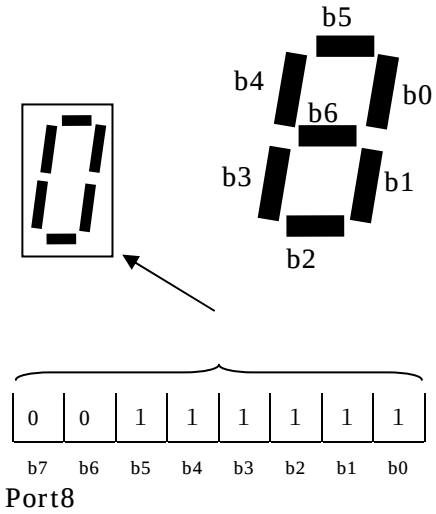
ポート８に値を出力すると、ビットが１の場合ＬＥＤが点灯し、０の場合消灯します。



7セグメント(SEG7)

ポート８に値を出力することで、その値に応じた値のＬＥＤセグメントが点灯するようになっています。

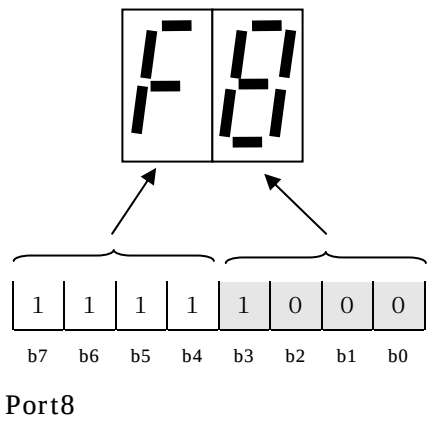
（例）０を表示したい場合は、b0,b1,b2,b3,b4,b5 を１に b6 を 0 にします。したがって、ポート８には６３（１６進数で３Ｆ）を出力すれば良いことになります。



7セグメント(SEG7D)

ポート８に値を出力すること、でその値に応じた値（０～１０、Ａ～Ｆ）が点灯するようになっています。

（例）F8 を表示したい場合は、b3 から b7 を１に、それ以外を０にすればよいことになります。



©2011 uaubn

DC モータ動作(DCMT)

ポート５に出力するデータにより、モータが動作します。上位４ビットのデータは無視します。

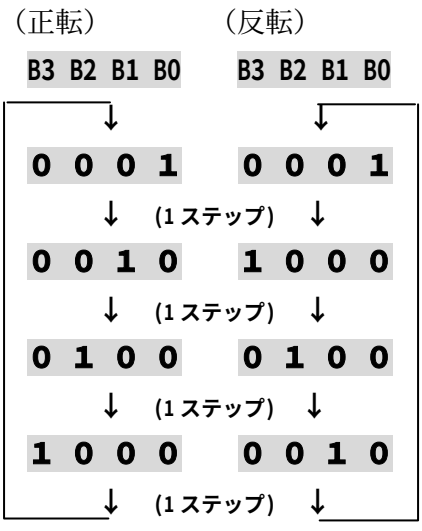
	B3	B2	B1	B0	動作
右モータ			0	0	停止
			0	1	正転
			1	0	反転
			1	1	強制停止
左モータ	0	0			停止
	0	1			正転
	1	0			反転
	1	1			強制停止

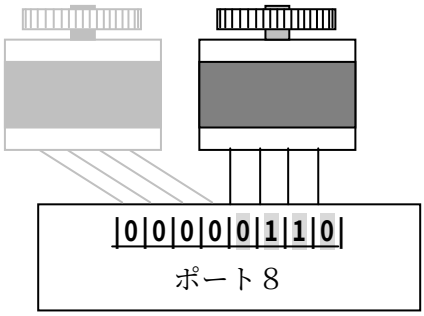
（例）下位４ビットに　０１０１　を出力　→　船前進

下位４ビットに　０１００　を出力　→　船右折

ステッピングモータ(STEPMT)

ステッピングモータには 4 つの入力（電源）線があり、回転させるには、その 4 つの入力に信号を連続して送ることで回転させます。ステッピングモータは、１ステップ毎の制御が可能です。



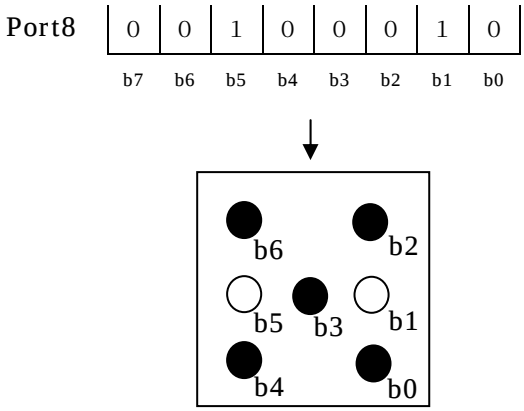


１ステップで１．８°回転します。

©2011 uaubn

サイコロ状ＬＥＤ(LED7)

ポート８に値を出力すると、ビットが
　１の場合ＬＥＤが点灯し、
　０の場合消灯します。
最上位ビットは使用していません。



©2011 uaubn

以下の命令はＨ８ＳＳ専用のＬＣＤ制御用の関数です。
実際のマイコンで使用する場合は別に関数を作成する必要があります。

ＬＣＤの表示

関数を呼び出すと自動的にＬＣＤ表示エリアが画面上に出てきます。なお、表示はキャラクタ（文字）単位となります。

（使用例）

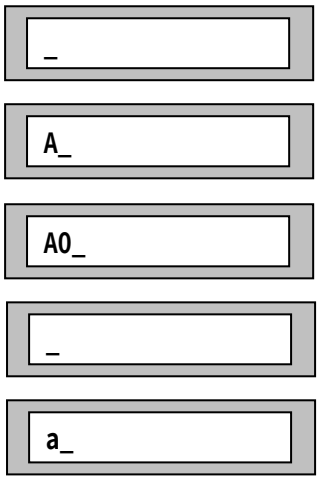
```
#include “3664.h”
void main(void)
{
   ClearScreen();                          (表示内容ｸﾘｱ)

   CharactorOut(‘A’);                      (’A’という文字を表示)

   CharactorOut(48);                       (文字コード 48(10)･30(16)の’0’を表示)

   ClearScreen();                          (画面ｸﾘｱ後カーソルは左端)

   CharactorOut(‘a’);
}
```



void CursorOn(void)

カーソルを表示します。

使用例

```
void func(void)
{
    CursorOn();        //  カーソルを表示します
}
```

void CursorOff(void)

カーソルを非表示にします。

使用例

```
void func(void)
{
    CursorOff();      //  カーソルを非表示にします
}
```


ⓒ2011 uaubn

void Locate(int x,int y)

カーソルの位置を桁位置 x 行位置 y で指定します。ＬＣＤ左上の座標は（０，０）です。

使用例

```
void func(void)
{
    int a=2,b=1;
    Locate(2,1);     //  カーソルの位置を X 座標 2，Y 座標 1 に設定
    Locate(a,b);     //                    〃
    Locate(a*2,1);   //  カーソルの位置を X 座標 4，Y 座標 1 に設定
}
```

void ClearScreen(void)

ＬＣＤ画面に表示されている文字を全てクリアします。

カーソル位置は上段の左端となります。

使用例

```
void func(void)
{
    ClearScreen();
}
```

void CharactorOut(char dt)

カーソルのある位置に文字コード dt の文字を一文字出力します。

一文字表示後のカーソル位置は、次の桁に移動します。

１６文字を越えた部分については表示が行われません。

使用例

```
void func(void)
{
    char c=0x41;
    CharactorOut('A');     //  カーソル位置に'A'を表示（文字としての記述）
    CharactorOut(0x41);    //            〃            （コードを記述）
    CharactorOut(c);       //            〃            （変数内の文字コード）
    CharactorOut(c+1);     //  カーソル位置に'B'を表示
}
```

void StringOut(char *dt)

カーソルのある位置にアドレス dt から始まる文字列を出力します。

使用例

```
void func(void)
{
    char s[]="test";
    CharactorOut("test");     //  カーソル位置に "test" を表示（文字列の記述）
    CharactorOut(s);          //            〃                  （文字列配列名）
    CharactorOut(s+1);        //  カーソル位置に "est" を表示

}
```


ⓒ2011 uaubn

void NumberOut(int number)

カーソルのある位置に整数値 number(-32768～32767)を出力します。

表示桁数は、表示文(数)字数と同じです。

使用例

```
void func(void)
{
    char n=123;
    NumberOut(123);     //  カーソル位置に 123 を表示（数値の記述）
    NumberOut(n);       //          〃              （変数の内容）
    NumberOut(n*10);    //  カーソル位置に 1230 を表示
}
```

void FormatNumberOut(float fnumber,int left, int right)

カーソルのある位置に数値 fnumber（int または float の範囲内）を整数部 left 桁、

小数部 right 桁の桁数指定で出力します。符号は整数部桁数に含まれます。

整数部の値の無い桁は空白が出力され、小数部の値は　指定桁＋１桁目　が

四捨五入された値となります。また、表示すべき整数部が、整数部の桁数指定値の left より

大きい場合は left の値によらず、整数部全桁が表示されます。

使用例

```
void func(void)
{
    char n=123.45;
    FormatNumberOut(123.45,4,1);  //  カーソル位置に△123.5 を表示（数値の記述）
    FomratNumberOut(n,4,1);       //          〃              （変数の内容）
    FomratNumberOut(n*10,4,1);    //  カーソル位置に 1234.5 を表示

}
                                      ※上記△は空白１文字分を意味します
```

void BNumberOut(int number,int keta)

カーソルの位置に整数値 number(-32768～32767)を指定の keta 数で２進数で出力します。

使用例

```
void func(void)
{
    char n=123;
    BNumberOut(123,8);     //  カーソル位置に 01111011 と 8 桁で表示（数値の記述）
    BNumberOut(n,8);       //          〃              （変数の内容）
}
```

void HNumberOut(int number,int keta)

カーソルの位置に整数値 number(-32768～32767)を指定の keta 数で１６進数で出力します。

使用例

```
void func(void)
{
    char n=123;
    HNumberOut(123,4);     //  カーソル位置に 007B と 4 桁で表示（数値の記述）
    HNumberOut(n,4);       //          〃              （変数の内容）
}
```


©2011 uaubn